REMARKS

Status of the Claims

- Claims 1-25 are pending in the Application.
- Claims 1-5, 11-13, and 17-18 are elected under the restriction requirement.
- Claims 6-10, 14-16, and 19-25 are currently withdrawn.
- Claims 1-5, 11-13, and 17-18 are rejected by the Examiner.
- Claim 1 is currently amended.

Election of Claims

Applicant confirms the election of Species I having Claims 1-5, 11-13, and 17-18 under the three way restriction requirement described on page 2 of the present Office Action dated 3/8/06. Accordingly, Claims 6-10, 14-16, and 19-25 are currently withdrawn from consideration.

Applicant disagrees that there is no generic claim. However, Applicant understands that upon the allowance of a generic claims, consideration will be given to species claims which depend from or otherwise require all the limitations of an allowable generic claim per 37 C.F.R. 1.141.

Claim Objections

Claim 1 is objected to for an error in punctuation in line 4. Applicant has amended Claim 1 to overcome the objection. Applicant respectfully requests withdrawal of the objection to Claim 1.

Claim Rejections Pursuant to 35 U.S.C. §101

Claims 17-18 are rejected under 35 U.S.C. §101 because the claims are directed to non-statutory subject matter. Specifically, the Examiner notes that the present specification includes communications media in the definition of computer readable media in paragraph 0041. Applicant has amended the specification to separate the definition of communications media from that of computer-readable media. Applicant notes that the definition of

communications media remains but is now not within the defined scope of Claims 17-18. Applicant respectfully requests withdrawal of the 35 U.S.C. §101 rejection of Claims 17-18.

Claim Rejections Pursuant to 35 U.S.C. §103 (a)

Claims 1-5, 11-13, and 17-18 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over "Efficient Transparent Recovery in Client-Server Information Systems" by Lomet et al. in view of U.S. Patent No. 5,752,026 to Fortier. Applicant respectfully traverses the rejection because neither reference teaches a method of performing a snapshot query during a database recovery operation.

Lomet et al. teaches an integrated method for database server recovery and transparent application recovery in a client-server system. The approach, based on application message logging, is similar to earlier work on distributed system fault tolerance. However, Lomet et al. exploits advanced database logging and recovery techniques and request/reply messaging properties to significantly improve efficiency. Forced log I/Os, frequently required by other methods, are usually avoided. Restart time, for both failed server and failed client, is reduced by checkpointing and log truncation. The method ensures that a server can recover independently of clients. A client may reduce logging overhead in return for dependency on server availability during client restart. (Abstract).

Claim 1 recites:

> A method of querying a database during database recovery, the method comprising:
> reading a log file containing log file transactions to find incomplete transactions;
> comparing the log file transactions with transactions reflected in the database to find unentered transactions;
> inserting the unentered transactions into the database;
> asserting a first lock type on a data item associated with each incomplete transaction;
> removing a transaction reflected in the database if the transaction is not committed in the log file, wherein the removal of the transaction de-asserts a first lock type on a data item associated with the incomplete transaction; and
> permitting a snapshot query of the database concurrent with the removal of an incomplete transaction, wherein if a first lock type is detected on a first data item, the snapshot query is delayed until the first lock type is de-asserted.

Page 5 of the present Office action dated 3/8/06 indicates that paragraph 1 of page 466 of Lomet et al. cites the element "inserting the unentered transactions into the database".

Applicant respectfully disagrees. Paragraph 1, page 466, of Lomet et al. discusses a rollback request and execute again operation of a failed server. Lomet et al. states in paragraph 1 that "Thus the server can undo incomplete requests and re-execute them all as new requests with different interleaving and effects." Thus, Lomet et al. associates a rollback with an undo operation (removal operation) and then re-executes the removed request.

Applicant submits that this operation is distinct from the action of "inserting the unentered transactions into the database" as recited in Claim 1 because the method of Claim 1 does not involve a rollback (undo or removal) of a transaction or request before inserting unentered transactions. In this instance, Lomet et al. teaches a function opposite to that of the recited element of Claim 1 (insertion versus removal). Also, Lomet et al. addresses incomplete requests whereas the Claim 1 element addresses unentered transactions. Accordingly, Lomet et al. teaches a different method than that recited in Claim 1.

Page 5 of the present Office Action indicates that page 466 second paragraph of Lomet et al. teaches the Claim 1 element of "asserting a first lock type on a data item associated with each incomplete transaction".

Applicant respectfully disagrees. Lomet et al. teaches:

> "To force a reply message, the server flushes its log buffer in a single atomic write to the log. If the message lookup table still contains the corresponding request message (which must have the largest preceding MSN of the application), then this message can be discarded once the atomic log tile write is completed." (page 466, second paragraph).

Applicant notes that flushing (clearing) a log buffer is greatly different than asserting a lock type (adding information) on a data item with each incomplete transaction. In this instance Lomet et al. teaches a function opposite of that of the recited Claim 1 element (adding lock information versus flushing or clearing information). Accordingly, Lomet et al. teaches a different method than that recited in Claim 1.

Page 5 of the present Office Action indicates that pages 462-463, paragraph 2.1 of Lomet et al. teaches the Claim 1 element of removing a transaction reflected in the database

if the transaction is not committed in the log file, wherein the removal of the transaction de-asserts a first lock type on a data item associated with the incomplete transaction.

Applicant respectfully disagrees. Lomet et al. teaches:

"2.1 General Message Passing
Consider two processes that exchange a message, a sender and a receiver. Either or both may fail. We must ensure that the recovered states of the two processes either both contain the message exchange or that neither does. This amounts to requiring the atomicity of a send-receive interaction and explains why the most successful prior approach has been based on persistent message queues (see Section 1.2). However, queues require two distributed transactions for each message exchange, and hence two instances of two phase commit. One transaction is between the sender and the queue manager (so that the sender's state is advanced if and only if the request is persistently enqueued), and a second between the receiver and the queue manager dequeuing the message. If the second transaction fails, the message is returned to the queue, and the surrounding TP monitor guarantees that the receiver eventually retries the dequeuing. The net effect is that the state transitions of both the sender and the receiver form one atomic unit.
The drawback of the queue-based solution is its high 110 cost. Sender, receiver, and queue manager all force information to stable storage. Reducing this cost is one of the main points of this paper. As our algorithm demonstrates, it is sufficient that only the sender force-log the message. Then, if the receiver fails after having received the message but without making it stable, the receiver can again obtain the message from the sender, and this argument is essentially symmetric in the two roles of a process. In fact, this is obviously the weakest logging requirement that can satisfy the correctness criterion. However, this approach has certain non-obvious implications:
1. The sender takes the responsibility for recreating the message when the receiver fails and needs to replay the message exchange. The protocol also needs to consider when the sender can discard the logged message and hence the server needs to know when the receiver no longer has to re-obtain the logged message.
2. The receiver's recovery depends on the sender's having forced logged the message. Should the receiver fail after the message exchange, it cannot perform independent restart. It must communicate with the sender because the message exchange must be replayed, but the receiver itself may have no stable record for it.

> Our goal is to improve upon sender logging for the important case of request/reply message pairs. We specialize in the rest of the paper to client-server information systems where we can make specific assumptions about the roles of the processes in order to design an efficient protocol." (paragraph 2.1, pages 462-463).

Applicant notes that paragraph 2.1 of Lomet et al. discusses the high I/O cost of queue based approaches and that a lower I/O cost can be achieved if only the sender force-log a message. However, Applicant can find no teaching wherein the removal of the transaction de-asserts a first lock type on a data item associated with the incomplete transaction as recited in Claim 1. Applicant can find no indication in the cited portion that indicates that Lomet et al. discusses de-asserting a lock associated with an incomplete transaction after an uncommitted transaction is removed. Lomet et al. fails to teach this Claim 1 element.

Page 5 of the present Office Action indicates paragraph 2.1 teaches the Claim 1 element of permitting a snapshot query of the database concurrent with the removal of an incomplete transaction, wherein if a first lock type is detected on a first data item, the snapshot query is delayed until the first lock type is de-asserted. Applicant respectfully disagrees because paragraph 2.1 of Lomet et al. fails to discuss any snapshot query that is concurrent with the removal of an incomplete transaction.

Page 6 of the present Office Action indicates that Fortier at col. 2, 3, 5 and the abstract teach a snapshot query of the database.

Fortier teaches that concurrency control is used to ensure database consistency while allowing a set of transactions to *execute* concurrently (col. 2 line 66- col. 3 line 1). However, Applicant submits that Fortier does not explicitly teach permitting a snapshot query concurrent with the removal of an incomplete transaction. Removal of an incomplete transaction is different than *execution* of the transaction.

Applicant submits that Lomet et al. fails to teach the following Claim 1 elements:

> inserting the unentered transactions into the database;
>
> asserting a first lock type on a data item associated with each incomplete transaction;
>
> removing a transaction reflected in the database if the transaction is not committed in the log file, wherein the removal of the transaction de-asserts a first lock type on a data item associated with the incomplete transaction; and

permitting a snapshot query of the database concurrent with the removal of an incomplete transaction, wherein if a first lock type is detected on a first data item, the snapshot query is delayed until the first lock type is de-asserted.

Applicant submits the addition of Fortier does not cure the deficiency of Lomet et al. because Fortier also fails to teach these same Claim 1 elements. In addition, the combination of Lomet et al. and Fortier does not comprehensively teach the method of Claim 1 which recites specific steps to acquire a snapshot query of a database during the "recovery" of the database.

Applicant respectfully submits that neither Lomet et al. nor Fortier, either alone or in combination, teach a method to query a database during the recovery of a database. Specifically, the combination of Lomet et al and Fortier fail to teach the steps of inserting the unentered transactions into the database, asserting a first lock type on a data item associated with each incomplete transaction, removing a transaction reflected in the database if the transaction is not committed in the log file, wherein the removal of the transaction de-asserts a first lock type on a data item associated with the incomplete transaction, and permitting a snapshot query of the database concurrent with the removal of an incomplete transaction, wherein if a first lock type is detected on a first data item, the snapshot query is delayed until the first lock type is de-asserted as recited in Claim 1.

Since neither Lomet et al. nor Fortier teach all of the elements of the independent Claim 1, the combination cannot render this claim obvious. Applicant respectfully requests withdrawal of the 35 U.S.C §103(a) rejection of independent Claim 1 and its dependent Claims 2-5.

Applicant submits that independent Claims 11 and 17 include elements related to the query of a database during the recovery of the database and includes the aspect of performing a snapshot query concurrent with recovery operations which is absent in Lomet et al. and Fortier. As a result, the combination of Lomet et al. and Fortier cannot render obvious independent Claims 11 and 17 and their dependent claims because not all elements are present in the cited references. As a result, Claims 11-13 and 17-18 patentably define over the cited art. Applicant respectfully requests withdrawal of the 35 U.S.C. §103 (a) rejection of all pending claims.

Conclusion

Applicant respectfully requests reconsideration of all pending claims in light of the arguments above. Applicant respectfully requests a Notice of Allowance for all pending claims as they patentably define over the cited art.

Respectfully Submitted,

Date: June 7, 2006

Jerome G. Schaefer
Registration No. 50,800

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439